Programming model and application porting to the Dynamical Exascale Entry Platform (DEEP)

EASC 2013
April 10th, Edinburgh
Damián A. Mallón

The research leading to these results has received funding from the European Community's Seventh Framework Programme (*FP7/2007-2013*) under *Grant Agreement* n° 287530

Outline

- DEEP Architecture
- Programming Model
- Applications
  - Climate Simulation (CYI)
  - CFD (CERFACS)
  - Superconductivity (CINECA)
- Conclusions

Credit where credit is due

- DEEP is a collaborative effort with contribution from many partners
- Contribution to the work here presented
  - Jülich Supercomputing Centre
  - University of Heidelberg/EXTOLL
  - Par-Tec
  - Barcelona Supercomputing Centre
  - Cyprus Institute
  - CERFACS
  - CINECA

DEEP Architecture

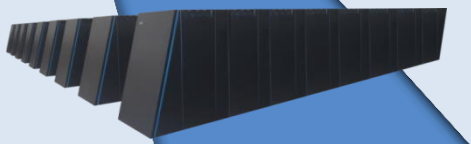
Constellation Systems

Intel Cluster Systems
(Juropa)

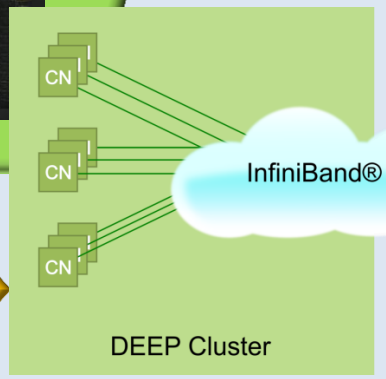
Graphic-card
Accelerated
Clusters

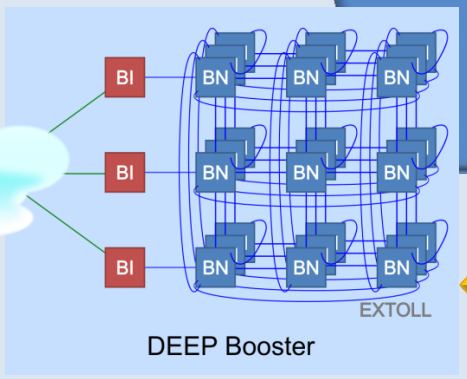
IBM Blue Gene/L

IBM Blue Gene/P



IBM Blue Gene/Q





HPC Cluster Architectures

Extremely Scalable Architectures

DEEP Architecture

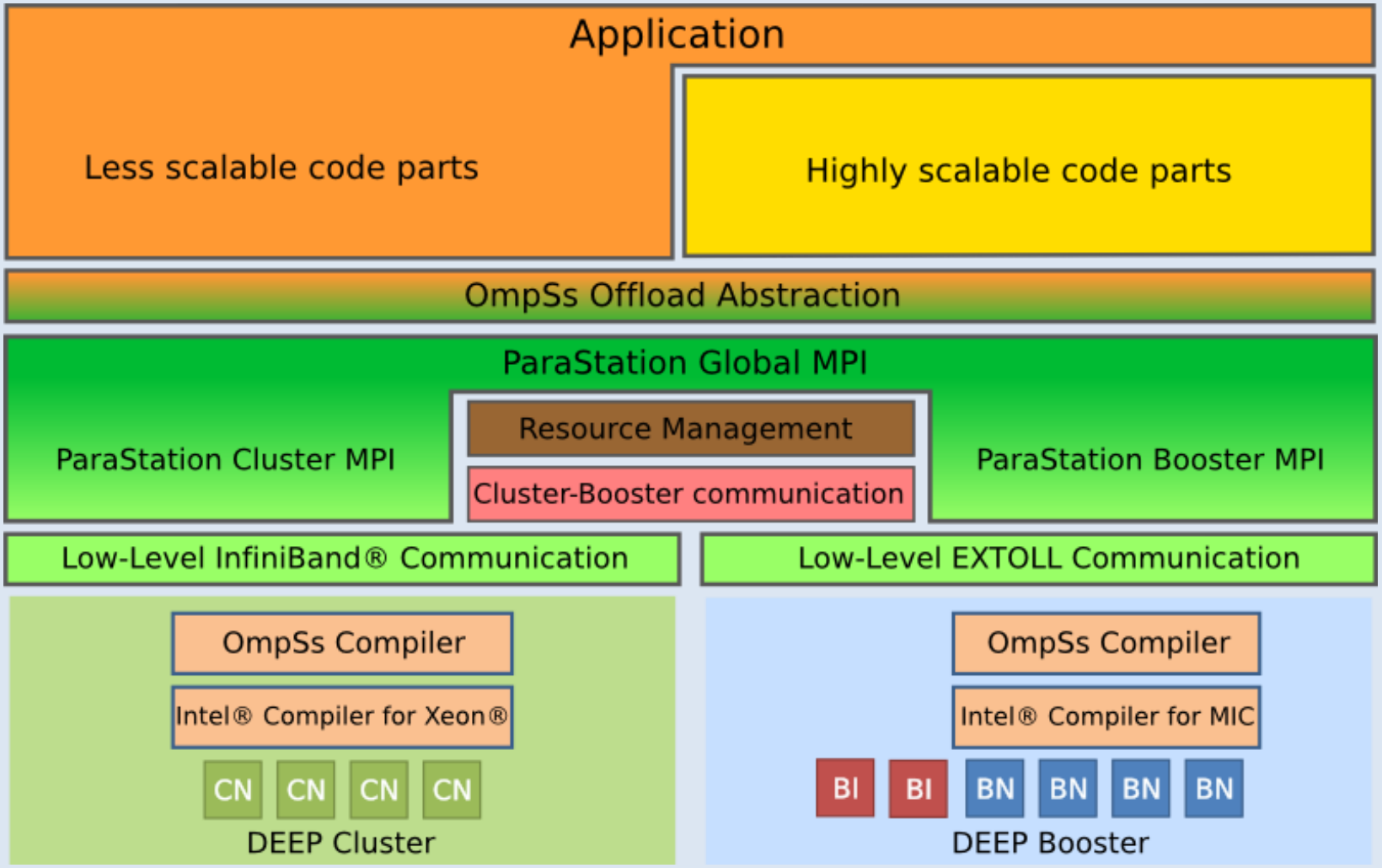
- Cluster Nodes
  - 128 Nodes
  - 2x Sandy Bridge processors
  - InfiniBand network
    - Fat tree
  - Connected to I/O
- Booster Nodes
  - 512 Nodes
  - 1x Xeon Phi
  - EXTOLL network
    - 3D torus
- Cluster-Booster Interface Nodes
  - 32 Nodes
  - InfiniBand and EXTOLL

DEEP Architecture

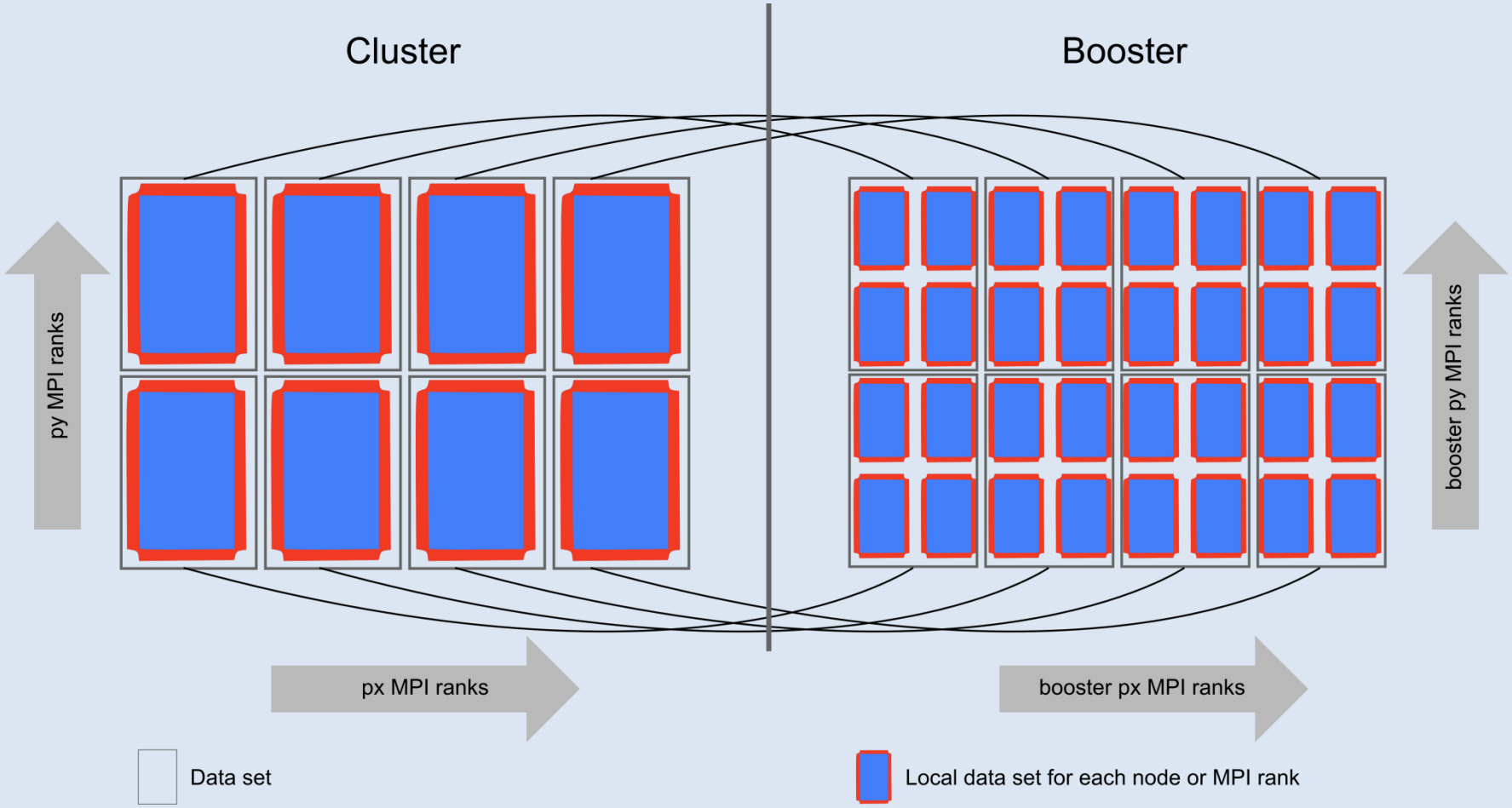
Technology	Implementation	Internal Clock Frequency	Link Bandwidth [Gb/s]	Latency [µs]	Effective maximum Bandwidth [GB/s]	Message Rate [msg/s]
EXTOLL VENTOUX	Xilinx Virtex6 FPGA	200 MHz	16 Gb/s	1.2 µs	Up to 1.4 GB/s	~ 25 millions
EXTOLL TOURMALET	65 nm ASIC	750 MHz	120 Gb/s	0.6 µs	Up to 10 GB/s	~ 100 millions
IB FDR*	45 nm ASIC	Unknown	56 Gb/s	0.8 µs	Not measured	Not measured
Typical 1GE	ASIC based	e.g. 125 MHz	1 Gb/s	e.g. 40 µs	0.11 GB/s	~0.5 millions
10GE	ASIC based	~125 to 312 MHz	12,5 Gb/s	12.5 µs	1.2 GB/s	< 2.5 millions
Cray Gemini	90 nm ASIC	650/800 MHz	75 Gb/s	1.5 µs	Up to 5.9 GB/s	~ 2 millions
Tianhe-1a	ASIC based	Unknown	80 Gb/s	2.5 µs	Up to 6.34 GB/s	~ 1-3 millions
TOFU (K-Computer)	65 nm ASIC	312.5 MHz	50 Gb/s	1.5 µs	Up to 4.76 GB/s	> 8 millions

* Mellanox literature data

Programming Model

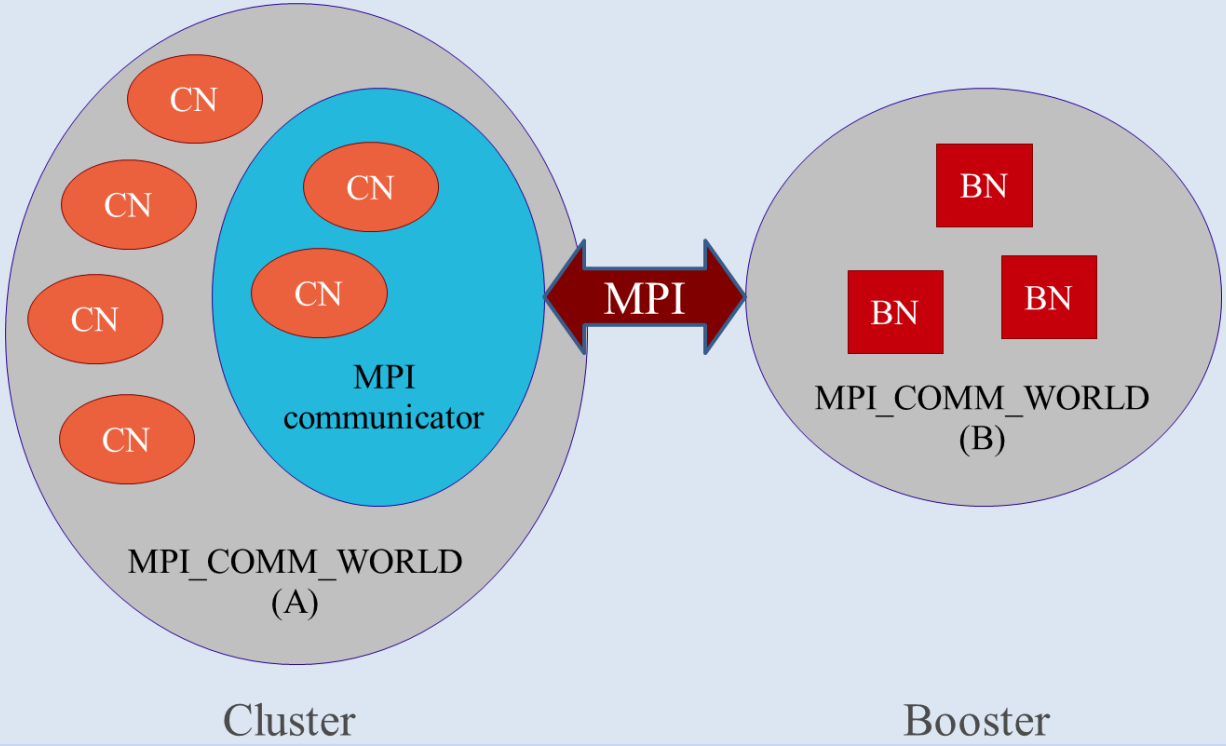


Programming Model



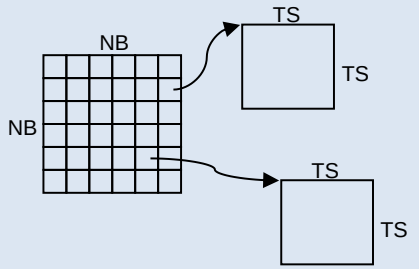
Programming Model

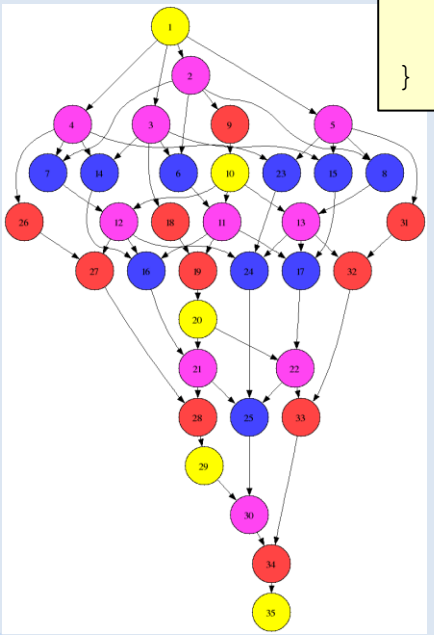
- "Global MPI"
  - Connect Cluster MPI and Booster MPI via MPI_Comm_spawn ()
  - Startup mechanism for Booster code parts
  - P2P communication mechanism via intercommunicator



Programming Model

```
void Cholesky( float *A[NT] ) {
int i, j, k;
for (k=0; k<NT; k++) {
   spotrf (A[k][k]) ;
   for (i=k+1; i<NT; i++)
      strsm (A[k][k], A[k][i]);
   for (i=k+1; i<NT; i++) {
      for (j=k+1; j<i; j++)
         sgemm( A[k][i], A[k][j], A[j][i]);
      ssyrk (A[k][i], A[i][i]);
   }
}
```



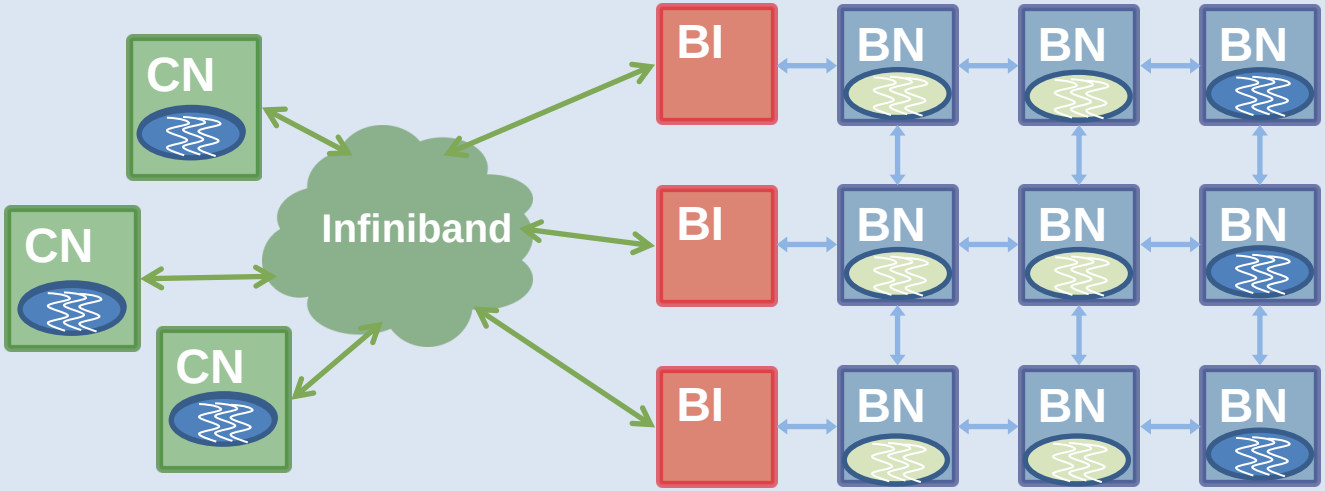


```
#pragma omp task inout ([TS][TS]A)
void spotrf (float *A);
#pragma omp task input ([TS][TS]T) inout ([TS][TS]B)
void strsm (float *T, float *B);
#pragma omp task input ([TS][TS]A,[TS][TS]B) inout ([TS][TS]C)
void sgemm (float *A, float *B, float *C);
#pragma omp task input ([TS][TS]A) inout ([TS][TS]C)
void ssyrk (float *A, float *C);
```

Decouple how we write (think sequential) from how it is executed

Programming Model

- Approach
  - MPI+OpenMP/OmpSs @ Cluster
  - MPI+OpenMP/OmpSs @ Booster
  - MPI+OpenMP/OmpSs @ whole System (Global MPI)



Programming Model

- “Collective” offload approach
  - Collective allocation of Booster Nodes
    - Define resources and communication name space

```
int DEEP_Booster_alloc( MPI_Comm cluster_comm,
                        int booster_nodes, ...,
                       MPI_Comm *intercomm);
```

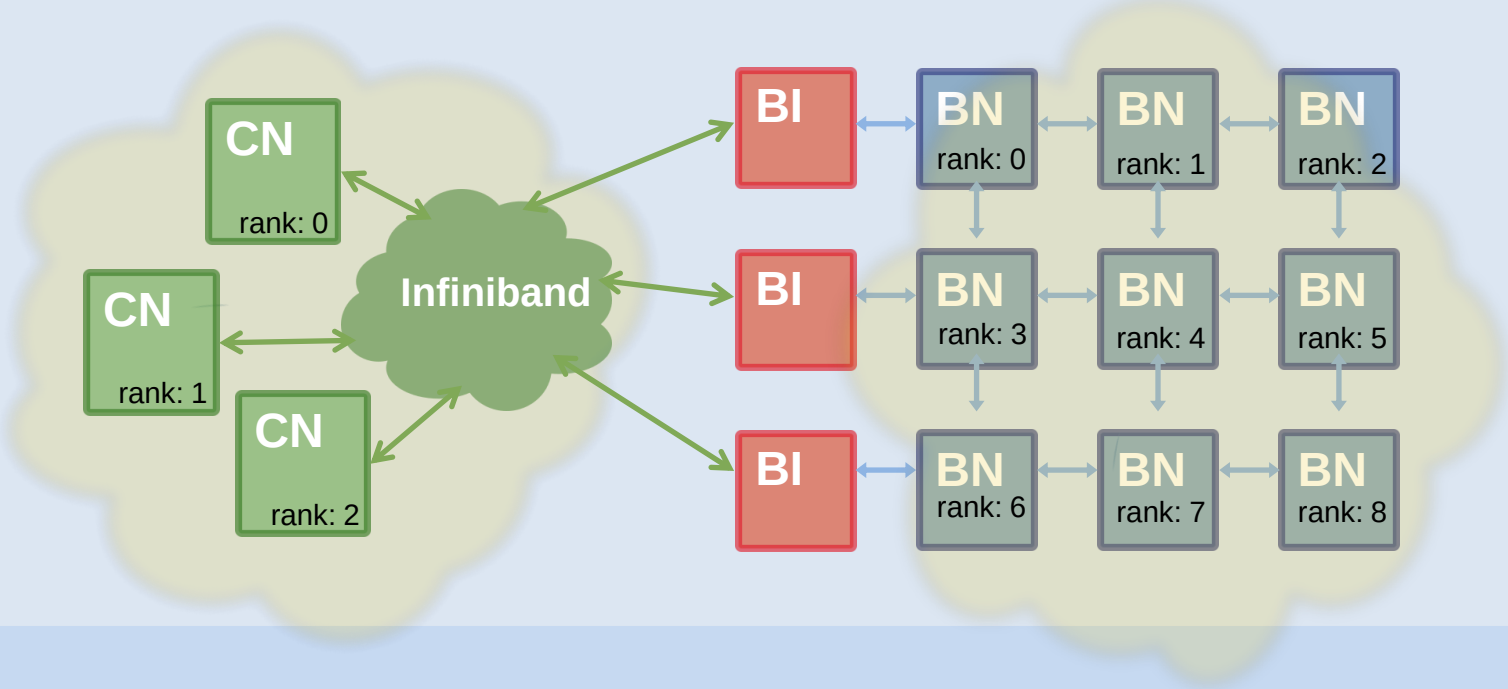
  - User controlled placement of tasks
    - Tasks can use MPI between them within allocated communicator

```
# pragma omp target device (intercomm:rank)
```

  - Standard MPI and OmpSs elsewhere
    - i.e.: CN ↔ BN transfers are asynchrony handled by standard OmpSs

Programming Model

```
int main(int argc, char *argv[]){
  MPI_Init(…);MPI_Comm_rank(…);MPI_Comm_size(…);
  MPI_Comm comm;
  DEEP_Booster_alloc(MPI_COMM_WORLD, size*3, size*3,
                     &comm);
  for(int i=0; i<3; i++){
    #pragma target device (comm:size*rank+i) copy_deps
    #pragma omp task input(…) output(…)
    foo_mpi(i,…);
  }
}
```



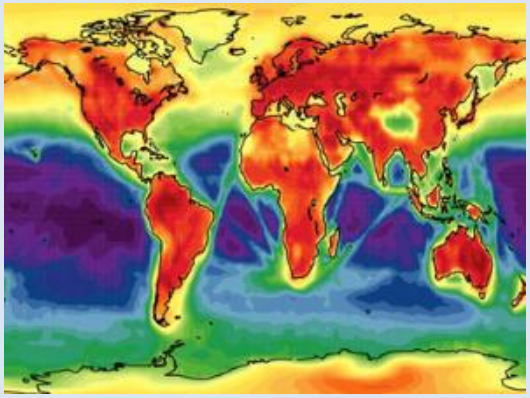
Programming Model

- Top-Level abstraction: OmpSs
  - **Hide the implementation details** from the application developer
  - Developer **annotates** the code to specify offloading semantics
- OmpSs allows to specify …
  - Booster topology (torus or grid, size)
  - Data distribution
- OmpSs compiler
  - Reads annotations, generates code calling the runtime
  - Generates (separate) executables for Cluster and Booster
- OmpSs runtime
  - Initiates kernels in the Booster
  - Handles data transfer between Cluster and Booster

Applications

- Climate Simulation (CYI)
- Computational Fluid Dynamics (CERFACS)
- High Temperature Superconductivity (CINECA)
- Seismic Imaging (CGGVeritas)
- Space Weather (KULeuven)
- Brain Simulation (EPFL)

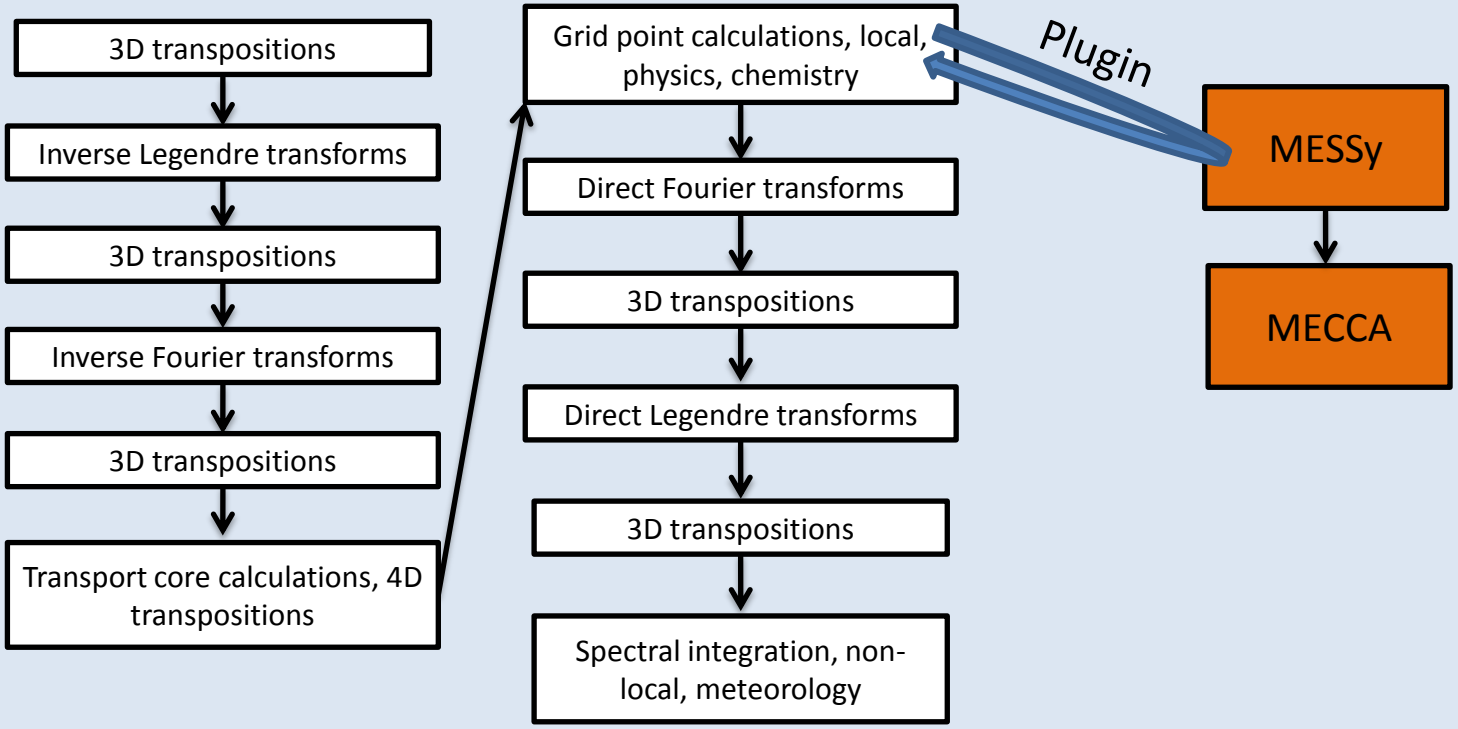
Climate Simulation (CYI)



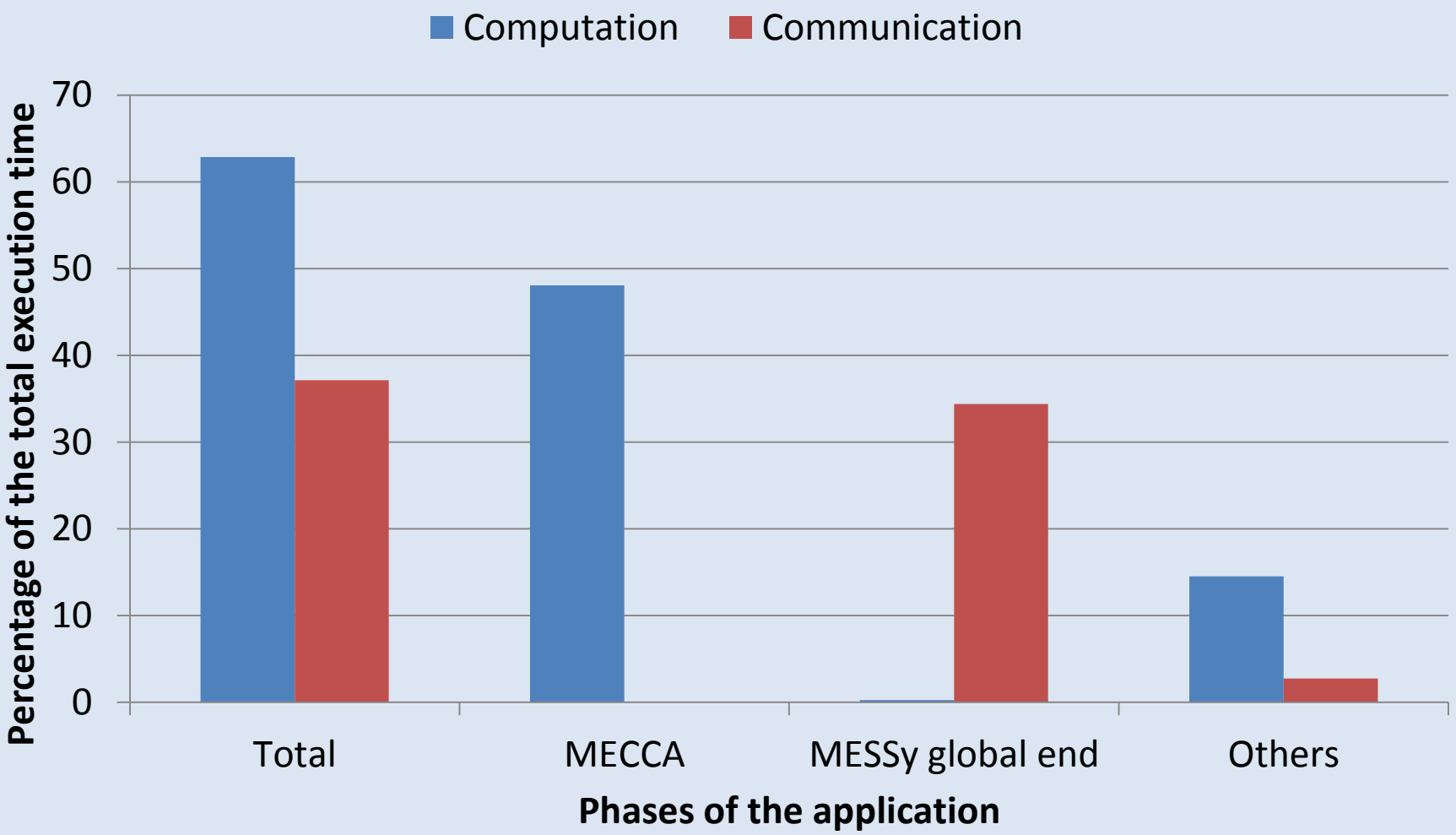
- Partner: CYI
- EMAC (ECHAM/MESSy Chemistry)
  - ECHAM provides a comprehensive general circulation model (GCM) of the atmosphere
  - MESSy (Modular Earth Submodel System) couples several physicochemical processes to it.
- Fortran 95

Climate Simulation (CYI)

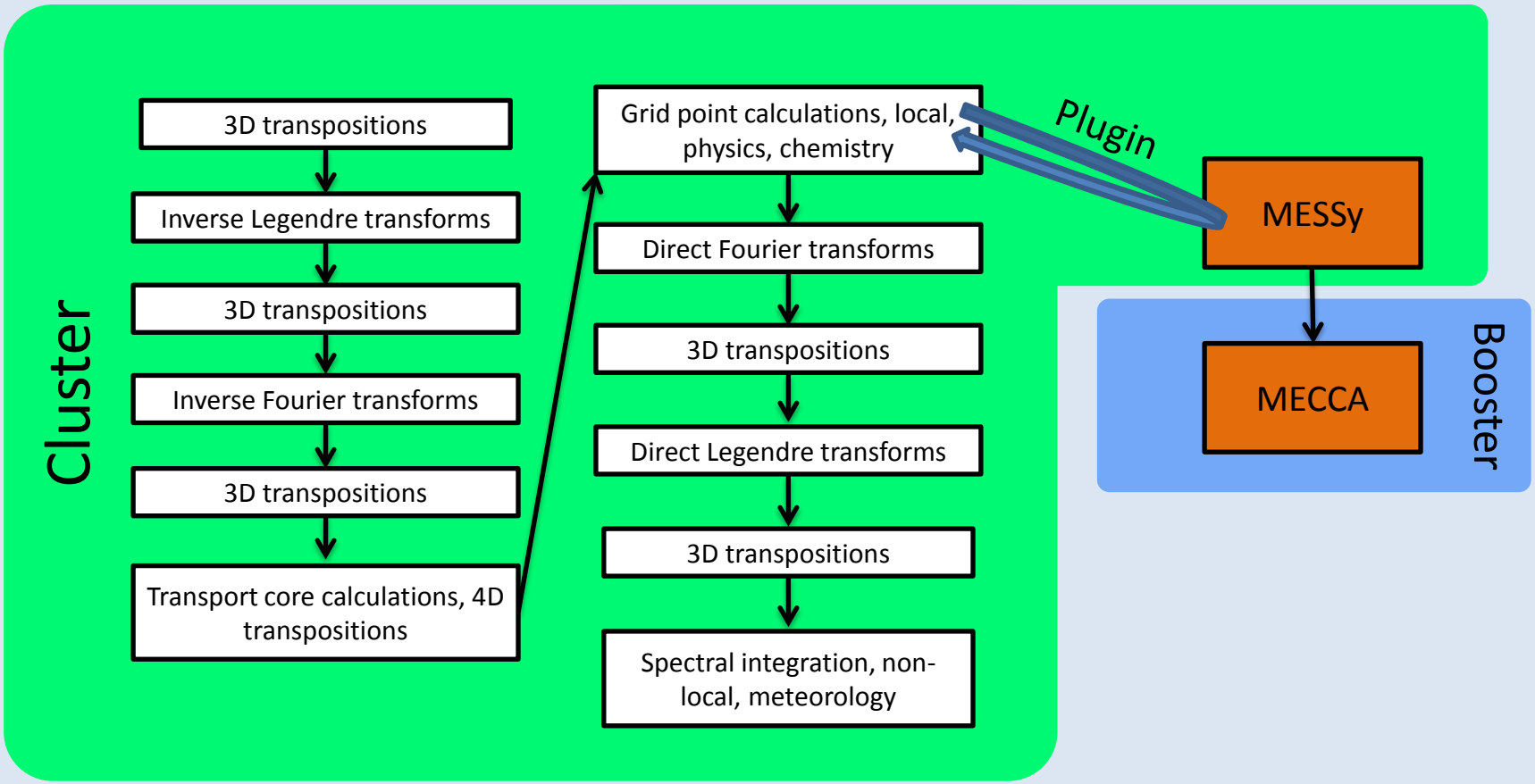
- Two coupled models:
  - ECHAM (Atmospheric)
  - MESSy (Physicochemical interactions)
  - EMAC = ECHAM/MESSy Atmospheric Chemistry



Climate Simulation (CYI)

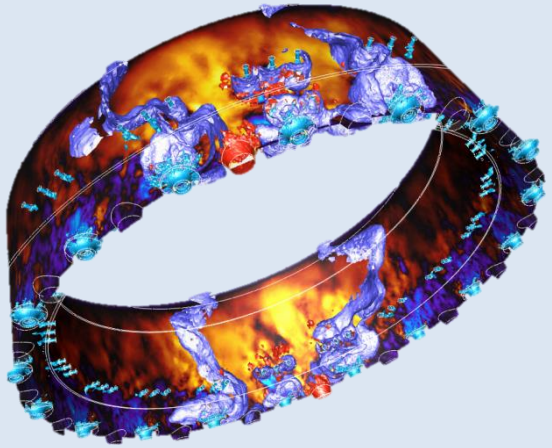


Climate Simulation (CYI)



CFD (CERFACS)

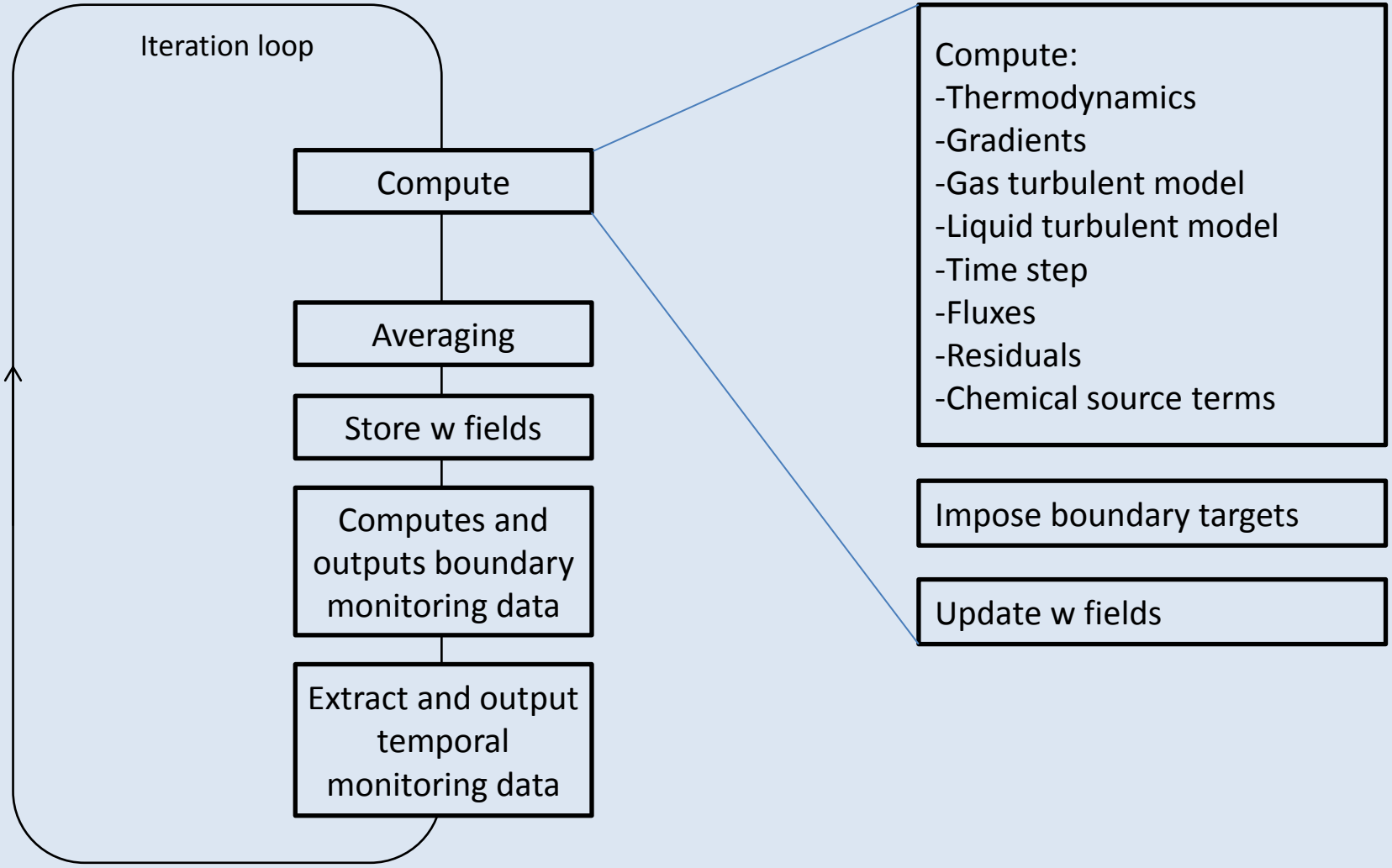
- Partner: CERFACS
- AVBP
- 3D Compressible Navier-Stokes solver
- Fortran + C



CFD (CERFACS)

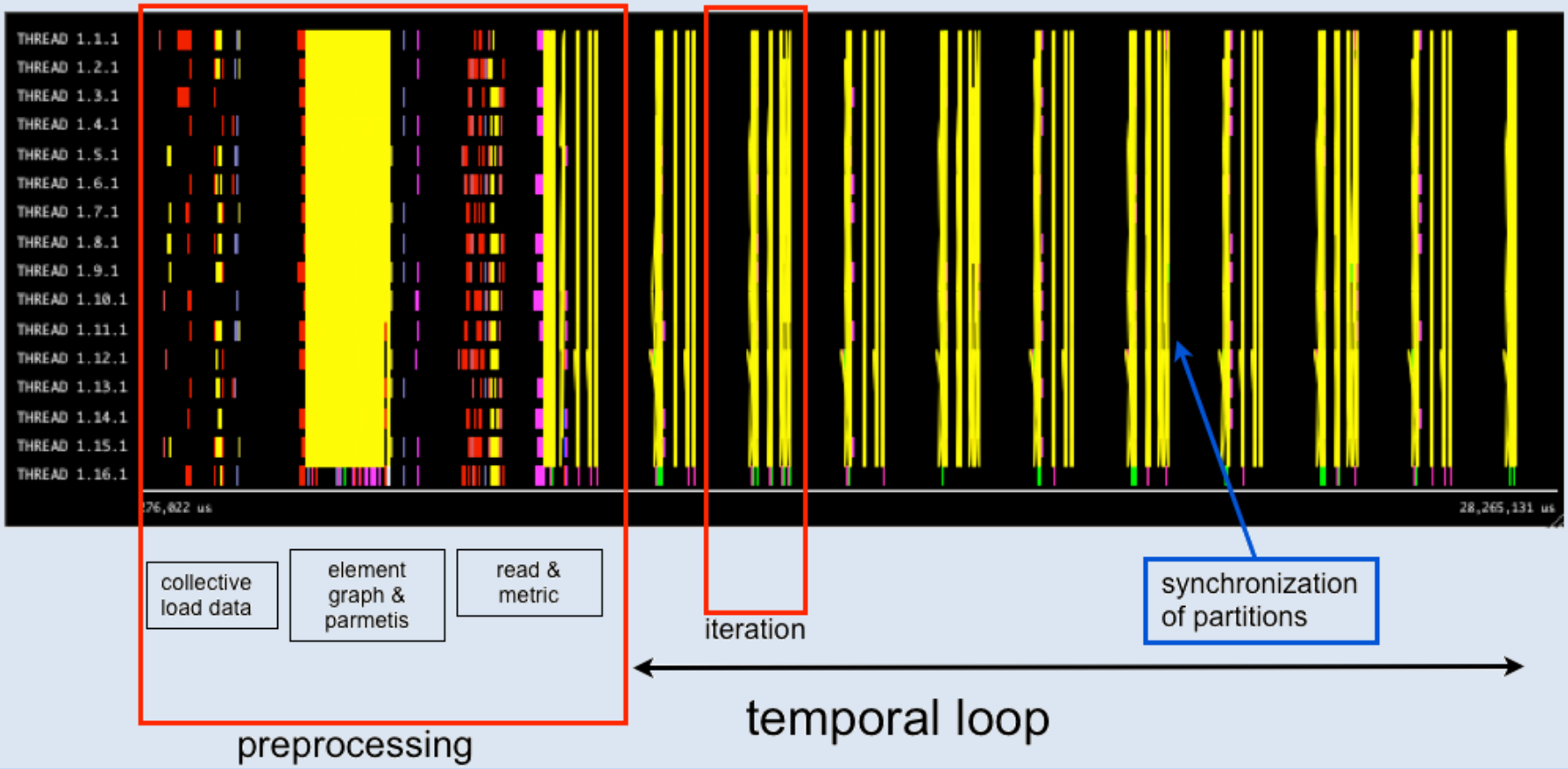
- Application structure
  - Phases
    - Read data
    - Decompose
    - Compute loop
      - Write snapshot every X cycles
  - The initial and I/O phases used to follow a master-slave approach
    - They have been optimized for scalability recently
  - Domain decomposition is done in parallel
  - I/O as well

CFD (CERFACS)


Iteration loop
Compute
Averaging
Store w fields
Computes and outputs boundary monitoring data
Extract and output temporal monitoring data
Compute:
-Thermodynamics
-Gradients
-Gas turbulent model
-Liquid turbulent model
-Time step
-Fluxes
-Residuals
-Chemical source terms
Impose boundary targets
Update w fields

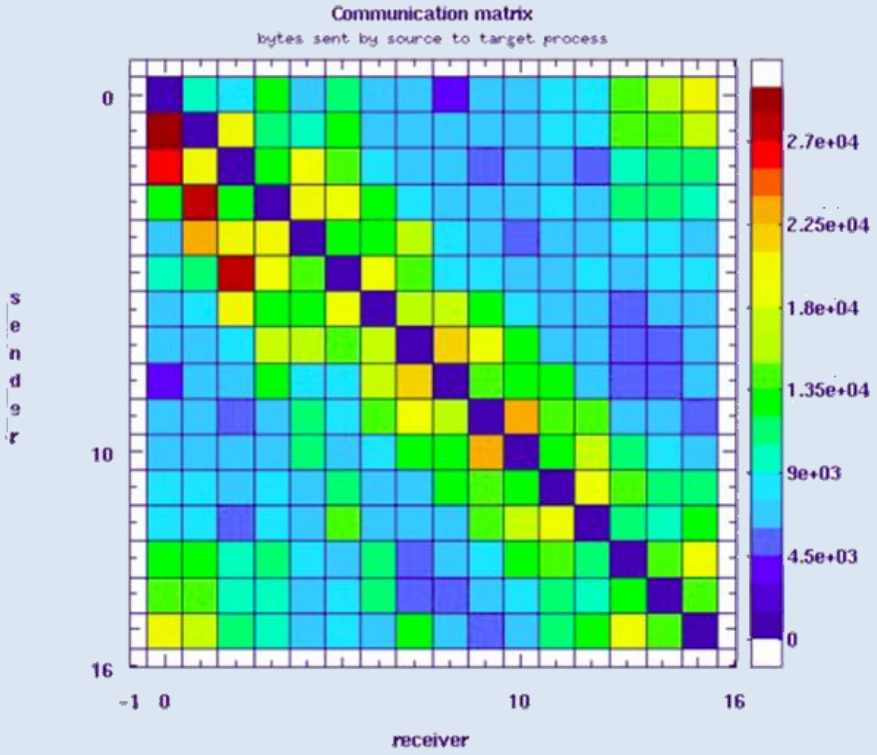
CFD (CERFACS)

- Trace of a 16 core simulation

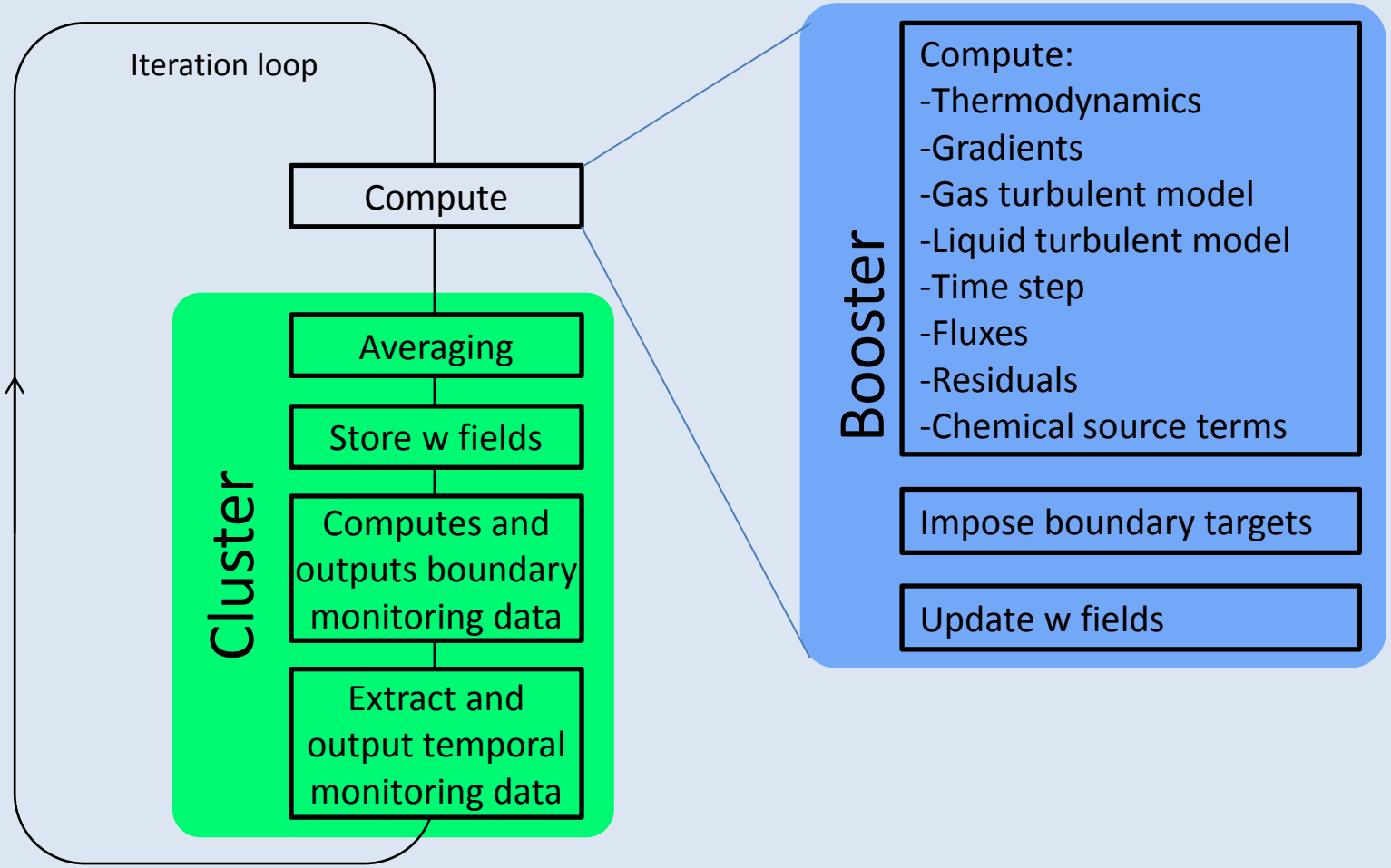


CFD (CERFACS)

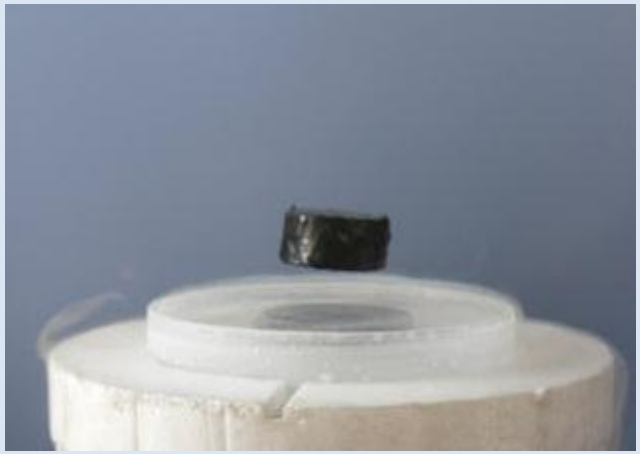
- Trace analysis (old version)
  - Computation takes ~50% of the time
  - Synchronization and boundary exchange takes ~20% of the time



CFD (CERFACS)


Iteration loop
Compute
Cluster
Averaging
Store w fields
Computes and outputs boundary monitoring data
Extract and output temporal monitoring data
Booster
Compute:
-Thermodynamics
-Gradients
-Gas turbulent model
-Liquid turbulent model
-Time step
-Fluxes
-Residuals
-Chemical source terms
Impose boundary targets
Update w fields

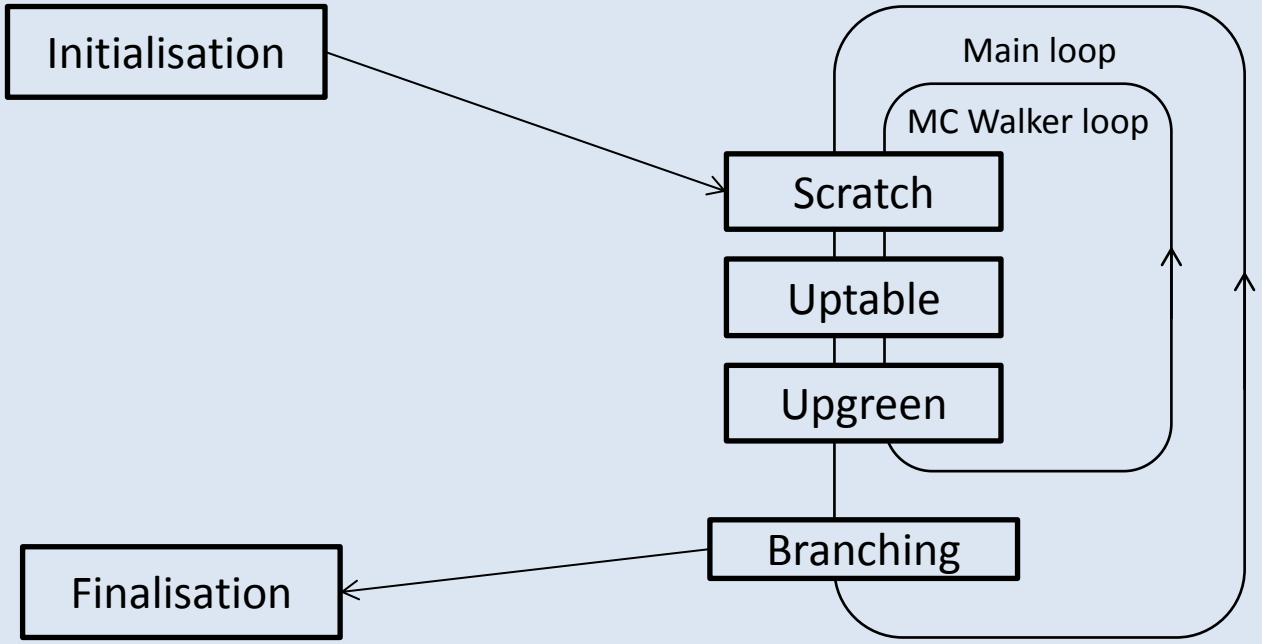
Superconductivity (CINECA)



- Partner: CINECA
- TurboRVB
- Quantum Monte Carlo simulations to study superconducting mechanisms in high critical temperature materials
- Fortran 90

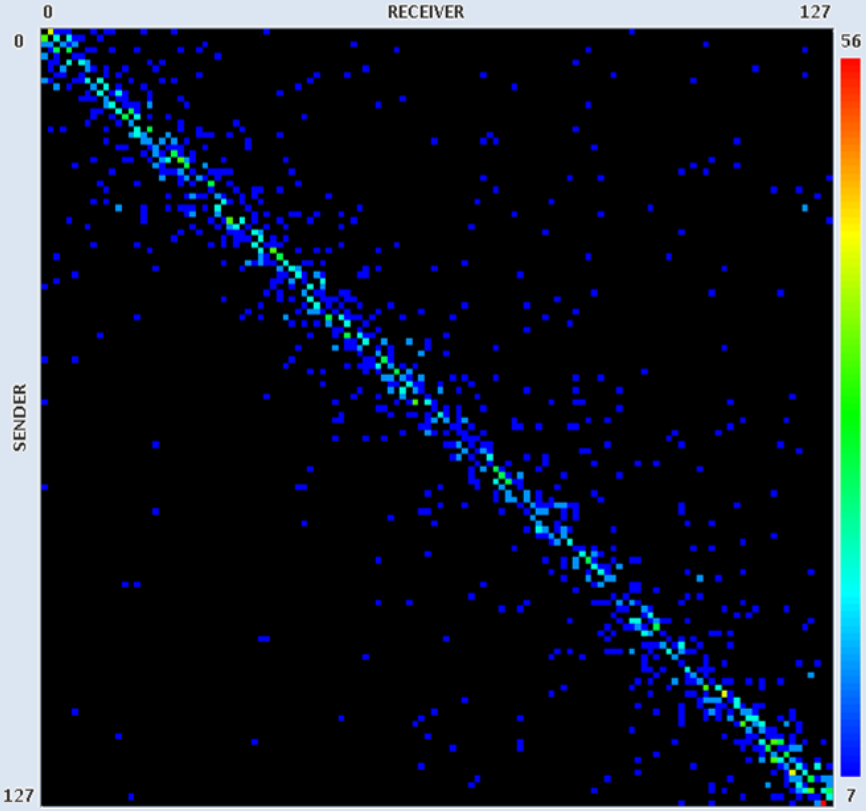
Superconductivity (CINECA)

- Phases:

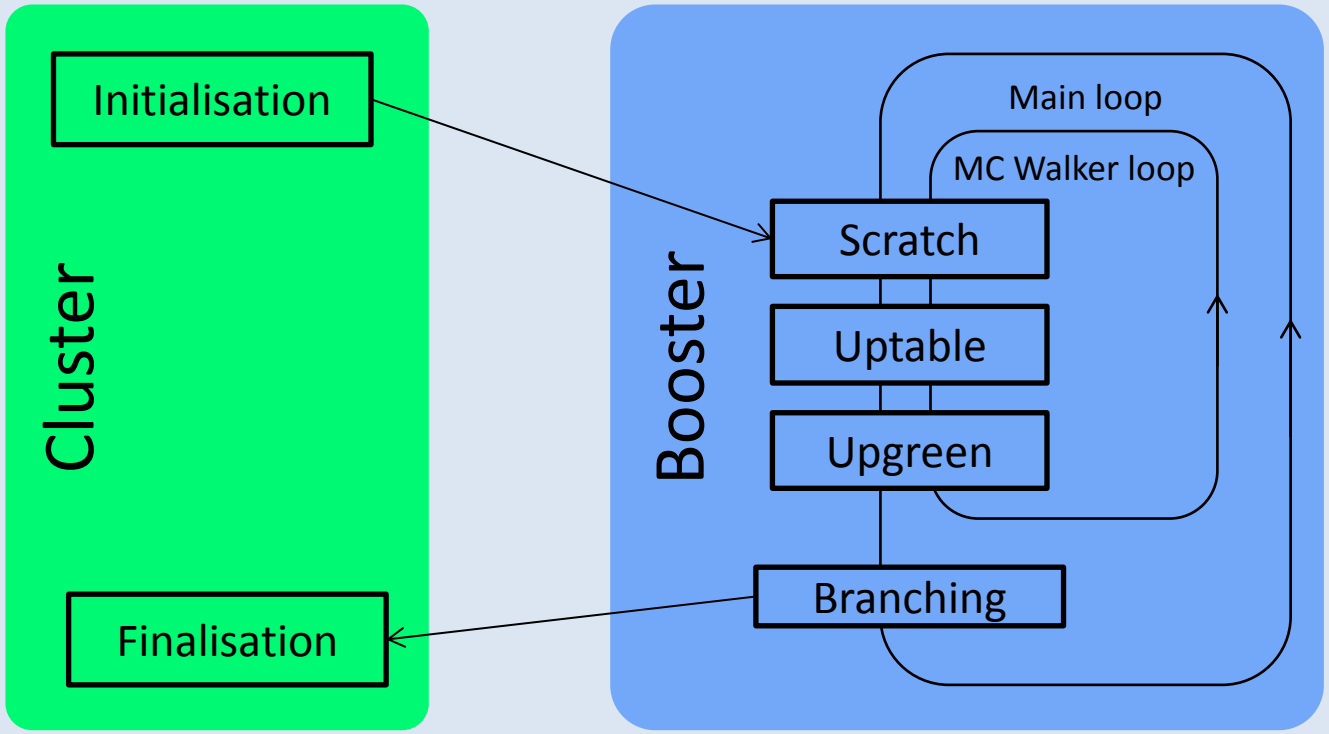


Superconductivity (CINECA)

- TAU communication matrix (number of calls)
  - Mainly nearest neighbours communication



Superconductivity (CINECA)



Conclusions

- DEEP provides a novel and flexible platform
- The programming model uses well known APIs
  - Plus a few changes to leverage asynchronicity and easy offloading
- Applications will be ported to it
  - Division between Cluster and Booster is clear now

Thank you